A True Random Number Generator for Crypto Embedded Systems

Norashikin M. Thamrin[1], Illiasaak Ahmad[1], Mohamed Khalil Hani[1]

[1]VLSI-ECAD Research Laboratory,Microelectronic and Computer Engineering Department (MiCE)
Faculty of Electrical Engineering,Universiti Teknologi Malaysia,81310 Skudai, Johor, Malaysia.
Tel: +607-5535223, Fax: +607-5566272, E-mail: norashikin.mthamrin@gmail.com,
ilyasak_ahmad@yahoo.com, khalil@fke.utm.my

Abstract

In this paper, we design a true random number generator (TRNG) in hardware which is targeted for FPGA-based crypto embedded systems. All crypto protocols require the generation and use of secret values that must be unknown to attackers. Random number generators (RNG) are required to generate public/private key pairs for asymmetric algorithm such as RSA and symmetric algorithm such as AES. Since security protocols rely on the unpredictability of the keys they use, RNGs for crypto applications must meet stringent requirements. The most important in cryptography is that attackers must not be able to make any useful predictions about the RNG outputs. The TRNG employs internal analog phase-locked loop (PLL) circuitry to generate a noise which is useful in producing random output. In contrast with traditionally used free running oscillators, it uses a novel method of random noise extraction based on two rationally related synthesized clock signals. The digital design is extremely compact and can be implemented on any advance FPGA device equipped with analog PLL. With the help of this TRNG, the cryptographic implementations such as key generation, authentication protocols, digital signature schemes and even in some encryption algorithm can be secure enough from being abused by malicious act.

Keywords

True Random Number Generator, Synthesized Clock, PLL, Embedded system, Cryptography

1. Introduction

The issue of random number generator is becoming crucial for implementation of cryptographic systems in Field Programmable Gates Array (FPGAs). Key generation, authentication protocols, zero-knowledge protocols, and block padding challenges are some of the cryptographic objects where a string of unpredictable bits or random numbers are required. In all theses applications, security greatly depends on the quality of the source of the randomness. The quality of generated numbers is proved by statistical tests. In addition to good statistical properties of the obtained numbers, the output of the generator used in cryptography must be unpredictable. For this reason, pseudo-random number generators, which are deduced through algorithmic process, are not suitable for cryptographic applications.

TRNGs can be produced using any non-deterministic process. This means that special attention should be paid to avoid weaknesses that help the attacker to break a system. The design is aimed to be implemented in embedded crypto system. Digital circuits of modern Field Programmable Logic Devices (FPLDs) include only limited source of randomness, e.g. metastability, frequency of free-running oscillators, clock jitter, etc. Usually, reliable and fast TRNG based on metastability is hard to implement and not secure enough in cryptographic applications. Free running oscillators are typically used in known FPGA based TRNGs. In principle, TRNGs based on free running oscillator and intrinsic jitter contained in digital circuits can be used without any additional FPLD resources.

In this paper, we present a TRNG design of a novel method of randomness extraction based on two rationally related synthesized stable clock signals. It was shown that it is well suited for modern FPLDs with internal analog Phase-Locked Loop (PLL) circuitry (e.g. Apex, Cyclone or Stratix FPLDs from Altera).We use the large flexibility of PLLs embedded in Stratix FPLDs to demonstrate the relationship between PLL and TRNG configurations, the quality of the output random bit-stream, and the speed of the generator. Although the TRNG was developed for the Altera Stratix family of devices, the principle can be easily employed in other digital devices containing analog PLLs.

2. The PLL-Synthesized Clock Jitter

2.1. Analog PLLs embedded in digital circuits

A digital VLSI circuit uses advanced clock generation and distribution circuitry based on embedded analog PLLs. A simplified block diagram of one analog based PLL block available in advanced digital circuits is depicted in Figure 1. Each PLL block can provide at least one synthesized clock signal with frequency F_{OUT}

$$F_{OUT} = F_{IN}\,[m/(n*k)] = F_{IN}\,[K_M/K_D] \qquad (1)$$

where F_{IN} is the frequency of the external input clock source. Reference-, feedback- and post-divider values n, m and k can vary from one to several hundreds in FPGAs.

The Altera Stratix devices include two types of PLLs[7]:

- **Fast PLL (FPLL)**: Stratix devices include up to 8 FPLLs. The FPLLs offer general-purpose clock management with multiplication and phase shifting. The multiplication is implified in comparison to EPLL and uses only $m=k$ scaling factors with a range from 1 to 32. Input frequency can vary in dependency on m (for speed grade -5) from 15 to 717 MHz, output frequency from 9.4 to 420 MHz, and the frequency of the Voltage Control Oscillator (VCO) from 300 to 1000 MHz.

- **Enhanced PLL (EPLL)**: Compared to FPLL, EPLLs have additional configurable features like external feedback, configurable bandwidth, run-time reconfiguration, etc. The also have an enhanced range of parameters. The input frequency can vary for a speed grade -5 device from 3 to 684 MHz, output frequency from 9.4 to 420 MHz and the frequency of the VCO from 300 to 800 MHz. Reference-, feedback- and post-divider values n, m and k can vary from 1 to 512 (1024 for k) with a 50% duty cycle.

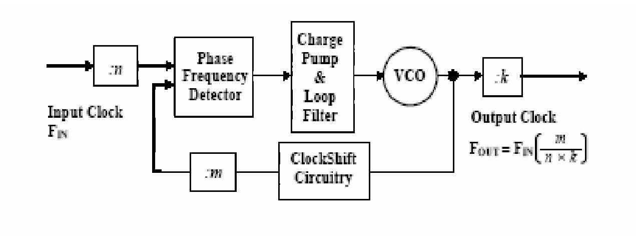

Figure 1 Simplified block diagram of an embedded PLL circuitry.

2.2. Jitter of PLL synthesized clock signals

In analog PLLs, a noise causes the Voltage Controlled Ocsillator (VCO) to fluctuate in frequency. Other frequency fluctuations are caused by variations of supply voltage, temperature and by noise environment. The internal control circuitry adjusts the VCO back to the specified frequency, but a certain part of the fluctuations caused by non-deterministic noise cannot be compensated for and is seen as a clock jitter. Clock jitter is the deviation from the ideal timing of clock transition events. As signals toggle faster and faster, the clock edge will fall outside the ideal margin. Figure 2 shows a schematic representation of clock jitter.

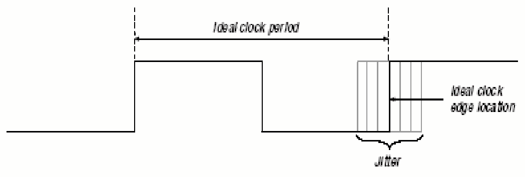

Figure 2: Clock jitter representation

2.3. Jitter generated in Stratix PLLs

The size of the intrinsic jitter depends on the quality factor Q of the VCO, on the bandwidth of the loop filter on the so-called pattern jitter introduced by the phase detector. The intrinsic jitter is often given in a peak-to-peak value or 1-sigma (RMS) value. The 1-sigma value of the jitter (σ_{jit}) depends on the technology and the configuration of the PLL and it can range up to 100ps. Since the technology of the PLL and the quality of VCO are usually defined, a user can change the output jitter directly by modification of scaling factors and filter bandwidth, but also indirectly by the design of the board (separation of the analog and digital ground, filtering of the analog power supply, etc).

Since the size of the jitter is very important in this method, previous researchers [1] have selected the Altera Development board with a Stratix EP1S25F780C5 device for jitter measurements and TRNG implementation. The jitter has been measured similarly using Agilent Infiniium DCA 86100B wide bandwidth oscilloscope. It has been found that for the FPLL with the ratio 12/7 the jitter achieves 1-sigma value of about 10 ps and for the EPLL with ratio 139/133 the 1-sigma value of the jitter is about 16 ps (see Figure 3(a) and 3(b)).

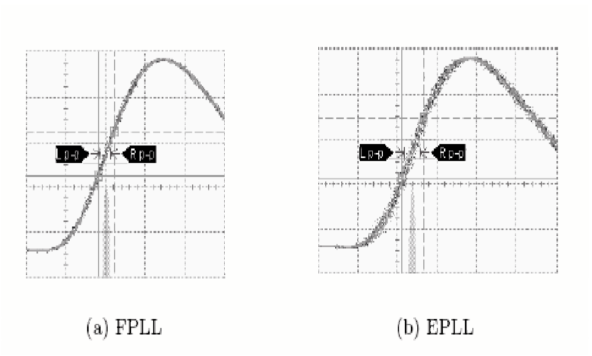

Figure 3: Jitter of the clock signal (horizontal scale: 200 ps/ div)

3. TRNG Architectures Embedded in FPGAs

The basic principle behind this method is to extract the randomness from the jitter of clock signal synthesized in the embedded analog PLL. The jitter is detected by the sampling of a reference (clock) signal using rationally related (clock) signal synthesized in the on-chip analog PLL. The fundamental problem lies in the fact that the reference signal has to be sampled near the edges influenced by the jitter. Figure 4 shows the basic structure of the TRNG.

The TRNG can be designed using one or two PLLs. Since the Stratix family contains two types of PLLs, the implementation EPLL is chosen since the jitter size is bigger than FPLL. The extraction technique presented is to

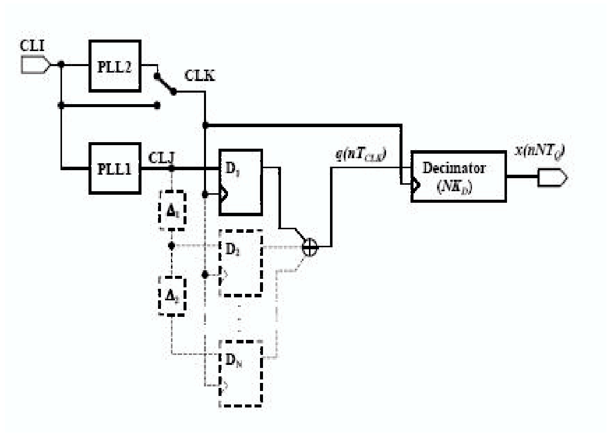

Figure 4: Basic structure of the TRNG [3]

use one clock signal to sample the value of a second clock signal on each cycle. If the two clock frequencies are slightly different, the point sampled in the second signal will advance through the second signal's cycle. If the change is small enough it will eventually sample the second signal in the jitter zone. Thus the sampling will produce a large number of deterministic bit and at least one uncertain bit taken in the jitter zone. XORing the deterministic value and the non-deterministic bit produces a single random bit.

The proposed design as shown in Figure 5 consists of sampling circuit, serial/parallel converter, register, status register and a control unit. It uses a system clock and a synthesized PLL clock (which is identical to the system clock) as the input of the sampler circuit.

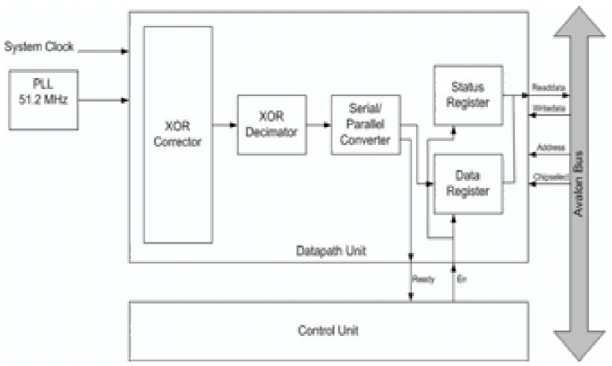

Figure 5: Overall design

The complete TRNG design is implemented on Altera Stratix EP1S40F780C5 development board. Acquired bits were transmitted to the PC through serial communication. The complete TRNG design which is including NIOS system module and serial interface controller needs up to 2,711 Logic Elements (LEs) from about 41250 LEs available in the device. The signal CLK was used as a clock signal for the control logic. The TRNG architecture was described in VHDL and implemented using Altera Quartus II development system, version 4.0. Because the jitter depends on an analog process, the real TRNG output cannot be simulated.

4. Experimental Results

The design and testing of this TRNG was done on Altera Stratix development board. Our VHDL development system is Window based. Our TRNG can produce up to 32000 bps. This paper presents results from the US National Institute of Standards and Technology (NIST) Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic applications.

The NIST test suite produces a summary report for each file of random bits it tests. Table 1 that follows is a result of running the NIST [10] suite over the set data produced by our TRNG. Each test in the NIST suite is run over a large number of sets of bits from the file to be tested. The statistic that is generated from each of these runs is called P-Value and it represents the probability of a perfect random number generator. According to NIST, if the P-Value that obtained from the test exceeds 0.001 (which is range from 0 to 1), the generator is acceptable in producing random bit. Thus, very small P-Values are bad. The documentation that accompanies the suite indicates that "If P-Value >=0.001, than the sequence can be considered random and uniformly distributed" [10]. There are 16 tests include in NIST documentation. However, the most common statistical test of TRNG is the Frequency (Monobit) test [2] [10]. In this test, our TRNG get P-Value = 0.841481 which is consider acceptable as a random number generator.

Table 1: NIST Results

NIST Test	P-Value
block-frequency	1.000000
cumulative-sums	0.499939
fft	0.771671
Frequency (monobit)	0.841481
linear-complexity	1.000000

Some of the suite tests were failed due to speed of random bit generation and limitation of bit generation. For future work, a better TRNG design can be implemented to speed up the random bit generation and can produce more random bits in one second.

5. Conclusion

This paper has described the methodology and design of PLL-based true random number generators embedded in modern FPLDs. The design uses the analog PLLs embedded in Altera Stratix FPGA family. The high quality of TRNG output was confirmed by applying special statistical tests.

The proposed solution is very cheap, uses few logic resources and is faster than comparable methods. Although the functionality of the proposed solution has been demonstrated for the Altera Stratix family, the same principle and design methodology can be used for all recent high-performance ASICs or FPLDs that include an on-chip reconfigurable analog PLL.

The generator is developed for embedded cryptographic applications which help to increase the system security but it can also be used in a wide range of other applications.

References

[1] Fischer V., et al. High performance true random number generator in Altera Stratix FPLDs. "Field-Programmable Logic and Applications," 14th International Conference, FPL 2004, Antwerp, Belgium, August 30-September 1, 2004, LNCS 3203, Springer, Berlin, Germany.

[2] Fischer V., et al. Simple PLL-based based true random number generator for embedded digital systems. 7th IEEE workshop Design and Diagnostic of Electronic Circuits and Systems, April 18-21, 2004, Stara Lesna, Slovakia.

[3] Drutarovsky M. et al. Implementation of hardware (true) random number generators (TRNGs) in reconfigurable devices. CryptArchi 2003, January 9-11, 2003, Saint-Etienne, France.

[4] Kohlbrenner P. and Gaj K., An embedded true random number generator for FPGAs. FPGA'04, February 22-24, 2004, Monterey, California, USA.

[5] Altera Technical Brief 70, Jitter Comparison Analysis: APEX 20KE PLL vs. Virtex-E DLL ver. 1.1, January 2001.

[6] Altera Application Note 115, Using the clocklock & clockboost PLL features in APEX devices ver 2.6, November 2003.

[7] Altera Application Note 282, Implementing PLL reconfiguration in Stratix & Stratix GX devices ver 2.0, December 2005.

[8] Altera Data Sheet: NIOS Development Board Reference Manual, Stratix Professional Edition ver 1.1, December 2003.

[9] Altera Application Note: Design debugging using signaltap II Embedded logic analyzer, June 2004.

[10] Rukhin A. et al., A statistical test suite for random and pseudorandom number generators for cryptographic applications, NIST Special Publication 800-22, May 15, 2001.